BAB II
LANDASAN TEORI

2.1 *Vehicle Routing Problem* (VRP)

Vehicle Routing Problem (VRP) dedefinisikan sebagai masalah dalam menentukan rute optimal dari satu atau lebih depot ke sejumlah retail yang tersebar secara geografis (Laporte, 1992). VRP digambarkan memiliki n-kendaraan yang memiliki kapasitas yang sama pada suatu depot yang melayani sejumlah konsumen atau retail. Depot tersebut membuat sejumlah rute yang akan dilalui kendaraan untuk mengirim barang ke sejumlah retail;. Pada VRP, perjalanan kendaraan diasumsikan berawal dan berakhir di depot yang sama. Selanjutnya dilakukan pemilihan rute dengan mempertimbangkan kapasitas kendaraan dan permintaan retail. Kemudian rute yang terpilih akan diurtukan agar menghasilkan jarak maupun waktu yang minimum.

VRP digambarkan sebagai sejumlah graf yang terhubung $G = (V, E)$ dimana $V = \{1, \dots, n\}$ merupakan himpunan tidak kosng dari simpul (*vertex* atau *node*) dan E adalah himpunan busur (*edges* atau *arcs*), $E = \{1, \dots, n\}$ yang menghubungkan sepasang simpul pada graf tersebut. *Vertex* atau *node* diartikan sebagai sejumlah retail yang akan dikunjungi, dimana v_1 diasumsikan sebagai depot. Sedangkan *busur* melambangkan jalan yang menghubungkan *retail i* ke *retail j*, $e = (v_i, v_j)$. Menurut (Gendreau et al., 1994) dalam merancang rute yang optimal untuk meminmasi biaya transportasi, VRP harus memenuhi batasan sebagai berikut:

1) Setiap rute kendaraan berawal dan diakhiri di depot
2) Setiap konsumen atau retail hanya boleh dikunjungi satu kali oleh satu kendaraan
3) Kendaraan yang digunakan adalah homegen dan memiliki kapasitas tertentu, sehingga permintaan pelanggan pada setiap rute yang dilalui tidak boleh melebihi kapasitas kendaraan.

2.1.1 Jenis-Jenis VRP

Studi mengenai VRP terus mengalami perkembangan mengikuti kondisi *real* di lapangan. Beberapa studi literatur membagi VRP menjadi beberapa jenis, seperti survei yang dilakukan oleh (Lin et al., 2014) sebagai berikut:

1) *Capacitated* VRP (1959)

   Capacitated VRP (CVRP) meruapakan masalah pengiriman barang menggunakan dengan mempertimbangkan kapasitas kendaraan yang digunakan. Pada CVRP kendaraan tidak boleh melakukan perjalanan apabila melebihi kapasitas angkutnya.

2) *Time-dependent* VRP (1966)

   Karalteristik yang dimilik Time-dependent VRP adalah waktu perjalanan antar noda bergantung pada jarak antar noda tersebut.

3) *Pickup and Delivery Problem* (1967)

   `pick up and delivery problem (PDP) diperkenalkan oleh Wilson and Weissberg pada tahun 1967. PDP menjanjikan pengiriman kembali setelah di penjemputan dilakukan.

4) *Multi-Depot* VRP (1969)

   Multi depot VRP (MDVRP) terdiri dari lebih dari satu depot dan masing-masing customer yang dikunjungi oleh satu kendaraan ditempatkan pada satudepot dari banyaknya depot yang akan dikunjungi.

5) *Stochastic* VRP (1969)

   Stochastic VRP terjadi apabila beberapa element seperti *customer demand, travel times,* bernilai acak. Teori probabilitas merupakan tool yang digunakan untuk menyelesaikan ketidakpastian tersebut dalam model matematika

6) *Location Routing Problem* (1973)

   Pengamatan yang menunjukkan bahwa lokasi depot yang terpisah dan rute kendaraan seringkali menghasilkan solusi suboptimal dan meningkatkan cost. Oleh sebab itu, Watson-Gandy & Dohm memperkenalkan Location Roting Problem (LRP). Keputusan bersama dalam LRP terdiri dari membuka satu dan membuat sejumlah rute untuk masing-masing depot yang dibuka.

Aplikasi LRP dapat ditemukan dalam lokasi postbox, pengiriman parcel, distribusi grosiran dll,

7) *Periodic* VRP (1974)

   Beltrami dan Bodin (1974) mengembangkan algoritma untuk menyelesaikan routing problems dengan batasan waktu, dimana lokasi customer membutuhkan jadwal kunjungan dan kombinasi hari yang berbeda dalam satu minggu.

8) *Dynamic* VRP (1976)

   Dynamic VRP berkonsetrasi pada dynamic operation dimana permintaan customer dikeluarkan selama periode perncanaan (online request) dan harus ditempatkan pada waktu yang sebenarnya untuk memastikan kendaraan yang cocok.

9) *Inventory routing problem* (1984)

   Inventory routing problem (IRP) pertama kali diperkenalkan oleh Bell et al (1983) yaitu integrasi antara manajemen inventori dan keberangkatan kendaraan.

10) *Multi-echelon* VRP (2009)

    Multi echelon VRP (MEVRP) mempelajari perpindahan aliran di dalam multi echelon distribution strategy. Pengiriman oleh angkutan dari tempat asal kepada customer harus dikirim melalui intermediate depot.

2.1.2 VRP *with fuel consumption and carbon emission*

Menurut Eglese and Bekta¸s, (2014) jumlah CO_2 yang dipancarkan oleh kendaraan kendaraan berasal dari proporsi jumlah bahan bakar yang dikeluarkan. Dalam literature, Eglese and Bekta¸s, (2014) menyebutkan ada dua cara untuk mengestimasi jumlah bahan bakar yang dipakai kendaraan: 1. *on-road measurement* yaitu perhitungan berdasarkan data *real time* dari emisi pada kendaraan yang sedang berjalan. 2. *Analytical fuel consumption* (*or emission*) models, dimana estimasi jumlah konsumsi bahan bakar berdasarkan pada jenis kendaraan, muatan, kecepatan kendaraan dan parameter lainnya.

Meneurut US Departement of Energy, (2008) dalam jurnal Kuo and Wang, (2011) terdapat beberapa faktor yang mempengaruhi jumlah konsumsi bahan bakar diantaranya:

1) *Weight of load*. Pada setiap kenaikan muatan 100 pound, jumlah konsumsi bahan bakar naik sebesar 2%.
2) *Speed of transportation.* Konsumsi bahan bakar biasanya meningkat dengan cepat pada kecepatan diatas 60 miles per jam. Pada kondisi jalan *highway, fuel consumption* lebih kecil daripada di perkotaan yang padat kendaraan.
3) *Use of air-conditioning*. Penggunaan *air-conditioner* (AC) dapat mengurangi *miles per gallon* (MPG) lebih dari 20%.
4) *Color of vehicle.* Pada siang hari, suhu permukaan kendaraan yang berwarna gelap lebih panas daripada kendaran yang berwarna terang. Hal tersebut akan mempengaruhi penggunaan AC.
5) *Inflation of tires.* Menajaga gesekan ban pada tekanan yang tepat dapat meningkatkan nilai MPG

Studi VRP mengenai faktor-faktor yang mempengaruhi *fuel consumption* banyak diselesaikan menggunakan metode metaheuristik seperti: Kuo, (2010) dan Xiao et al., (2012) menggunakan algortima *simulated annealing.* Penelitian tersebut mempertimbangkan *time-dependent travel speed* dan *fuel consumption rate* (FCR) untuk meminimasi konsumsi bahan bakar. Faktor *time-dependent travel speed* juga dibahas pada penelitian Kok et al., (2012), Duan et al., (2015), Qian and Eglese, (2016), Norouzi et al., (2017). Algortima yang dipakai antara lain adalah *Djikstra algorithm dan dynamic programming heuristic, ant colony algorithm, tabu search algorithm,* dan *particle swarm algorithm*.

Selain itu, Montoya et al., (2016) menentukan rute kendaraan dengan memeperhatikan alternative fuel vehicles (AFVs) untuk meminimasi *fuel consumption* dan karbon emisi. Montoya menggunakan algoritma *Multi-space sampling heuristic*. Poonthalir and Nadarajan, (2018a) melibatkan variasi kecepatan kendaraan berdasarkan konsep distribusi triangular ke dalam model matematis sistem yang dibangun. Penyelsaian VRP tersebut menggunakan

algortima *particle swarm optimization* (PSO). (Kuo and Wang, 2011) melakukan penelitian mengenai pengaruh berat muatan terhadap konsumsi bahan bakar menggunakan algoritma *tabu search.* Mengacu pada artikel Kuo and Wang, (2011), penelitian ini menggunakan konsep tersebut pada sebuah *case study* di perusahaan dengan algoritma yang berbeda yakni *particle swarm optimization* (PSO).

Untuk menyelesaikan masalah VRP dengan memepertimbangkan berat muatan,, Poonthalir dan Nadarajan, (2018) membuat dua fungsi tujuan, yaitu minimasi biaya transportasi (*route cost*) dan minimasi jumlah konsumsi bahan bakar (*fuel consumption*). Oleh karena itu, kedua peneliti tersebut menerapkan model *goal programming* untuk mendapatkan deviasi minimum pada kedua fungsi tujuannya. Adapun model matematisnya ditulis sebagai berikut:
Objek yang akan diminimasi:

2.1.3 Model Matematis VRP

Pada umumnya, VRP memiliki jaringan antar *customer* yang simetris dan bertujuan untuk meminimasi total jarak perjalanan kendaraan. Ada beberapa batasan (*constraint*) yang dipertimbangkan dalam model VRP diantaranya adalah; setiap customer dilayani sekali oleh satu kendaraan; setiap kendaraan berangkat dan kembali ke depot total permintaan untuk setiap rute kunjungan tidak melebihi kapasitas kendaraan dan waktu tempuh. Formulasi matematika yang ditulis pada penelitian ini mengacu pada artikel Yunita et al., (2013) . Agar relevan dengan kondisi pada studi ini, fungsi tujuan disesuaikan yaitu untuk minimasi konsumsi bahan bakar kendaraan.

Formulasi VRP dengan tujuan meminimasi *fuel* consumption pada sebuah *routing plan* adalah sebagai berikut:

a. $Minimize\ f = \sum_{v \in V} \sum_{r \in R^v} \frac{d_{ij}}{KPL_{ij}} \left(1 + p\left(L_{ij}/k\right)\right)_r^k x_r^v$ (1)

Persamaan (1) merupakan fungsi tujuan untuk meminimasi *fuel consumption*

b. $\sum_{k \in K} \sum_{r \in R^k} C_r^k x_r^k \geq 1 \qquad \forall\, i \in N$ (2)

Kendala pada pers (2) untuk memastikan semua pelanggan dapat dikunjungi.

c. $\sum_{k=1}^{K} y_{ik} = 1 \qquad \forall\, i \in V\ \{0\}$ (3)

Persamaan (3) menjamin setiap pelanggan dilayani satu kali oleh sebuah kendaraan.

d. $\sum_{k=1}^{K} y_{0k} = K$ (4)

Batasan pada pers (4) menajamin terdapat K kendaraan yang melakukan aktivitas distribusi

e. $\sum_{i \in V} d_i y_{ik} \leq C_k \quad \forall k = 1,2, \dots K$ (5)

Kendala pada persamaan (5) memastikan *demand* pelanggan pada setiap rute tidak melewati kapasitas kendaraan

f. $\sum x_r^k \geq 1 \quad \forall\, k \in K$ (7)

Kendala pada pers(7) ini memastikan setiap kendaraan minimal melewati satu jalur

g. $\sum_{r \in R^k} T_r^k X_r^k \leq T_{Max} \quad \forall$ k ∈ K (7)

Kendala pada pers (7) memastikan waktu tempuh kendaraan tidak melebihi waktu tempun yang diizinkan.

h. $x_r^k \in \{0,1\} \quad \forall\, k \in K,\ \forall\, r \in R^k$ (8)

Persamaan (8) merupakan kendala yang menjamin variabel keputusan x_r^k merupakan integer biner. Dengan variabel keputusan:

$$x_r^k = \begin{cases} 1, & jika\ kendaraan\ k\ menggunakan\ rute\ r \\ 0, & jika\ selainnya \end{cases}$$

Keterangan notasi :

KPL : jarak perjalanan per unit bahan bakar

p : kenaikan *fuel consumption* setiap penambahan mauatan sebesar 2%

k : kenaikan kapasitas muatan per 100 pound atau 45,35 kilogram

L_{ij} : beban muatan

K : himpunan kendaraan pada depot, dengan $K = \{1,2, \dots, k\}$

R^k : himpunan rute yang ditempuh kendaraan k, dengan $R = \{R^1, R^2, .., R^k\}$

V : himpunan titik

d_i : jumlah permintaan customer i

r : indeks rute

C_r^k : biaya operasioanal kendaraan k yang menempuh rute r

A_{ri}^k : konstanta, bernilai 1 jika kendaraan k dengan rute r yang menuju customer i dan bernilai 0 untuk kondisi lainnya

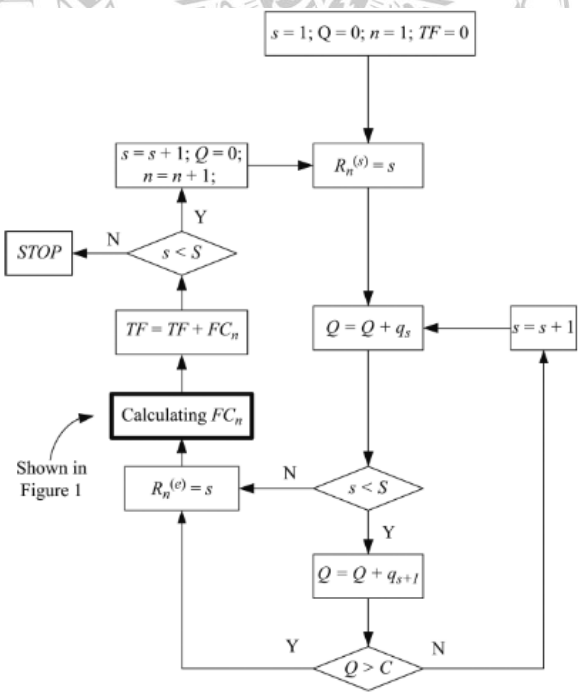
T_k^r : waktu yang dibutuhkan kendaraan menempuh rute r

T_{max} : waktu tempuh maksimum yang diizinkan

2.2 Prosedur Pembagian Rute

Kuo and Wang, (2011) dalam penelitiannya mengasumsikan bahwa kapasitas kendaraan terbatas. Sehingga sejumlah pelanggan yang akan dikunjungi dibagi menjadi beberapa sub-rute. Ilustrasi prosedur pembagian rute dapat dilihat pada gambar 2.1.

Prosedur pembagian rute terdiri dari beberapa tahap. Pertama, melakukan inisialisi dengan menentukan urutan rute secara acak. Kemudian tetapkan $s = 1$ urutan kunjungan pertama pelanggan. Tetapkan $R_n^{(s)} = s$ dimana $R_n^{(s)}$ adalah pelanggan pertama yang akan dikunjungi. Tetapkan $Q = 0$ yang menunjukkan total berat muatan kendaraan. Kemudian hitung jumlah permintaan pelanggan sebanyak q_s. Jika pelanggan pertama kurang dari total jumlah pelanggan atau $s < S$ maka sisipkan pelanggan selanjutnya ke urutan kunjungan kedua $s = s + 1$. Kemudian hitung $Q = Q + q_{s+1}$. Apabila total muatan kendaraan melebihi kapasitasnya $Q > C$ maka sub rute terbentuk dimana pelanggan yang terakhir dikunjungi $R_n^{(e)} = s$. Setelah itu lakukan prosedur perhitungan bahan bakar pada sub bab 2.3. ulangi langkah tersebut sampai seluruh pelanggan dilayani.



Gambar 2.1. Prosedur pembagian rute

2.3 Kalkulasi Konsumsi Bahan Bakar

Pada studi ini, model kalkulasi konsumsi bahan bakar merujuk pada penelitian yang dilakukan oleh Kuo and Wang, (2011). Berdasarkan pembahasan pada sub bab 2.1.2, jarak perjalanan, kecepatan kendaraan dan muatan kendaraan memeiliki pengaruh yang besar terhadap jumlah konsumsi bahan bakar. Dalam penelitiannya, Kuo dan Wang mengkalsifikasikan kecepatan kendaran menjadi 3 jenis – *high speed, medium speed, low speed*. *High speed* diasumsikan untuk transportasi pada *highway, low speed* diasumsikan untuk trasnportasi di perkotaan yang padat sedangkan medium speed antara kedua kondisi jalan tersebut.

Jika diketahui MPG_{ij} dan jarak d_{ij} antara retail i dan retail j, maka *fuel consumption* F_{ij}untuk kendaraan bermuatan kosong dari retail i dan retail j dapat dihitung pada persamaan (7)

$$F_{ij} = d_{ij}/MPG_{ij} \quad (9)$$

Selain itu, kenaikan muatan sebesar k pounds akan meningkatkan *fuel consumption* sebesar p persen. Jika kendaraan menuju retail i ke retail j dengan membawa muatan sebesar L, maka konsumsi bahan bakar yang sebenarnya dihitung pada persamaan (10)

$$FC_{ij} = \frac{d_{ij}}{MPG_{ij}}\left(1 + p\left(L_{ij}/k\right)\right) \quad (10)$$

Keterangan notasi:

F_{ij} = konsumsi bahan bakar tanpa muatan

MPG = jarak perjalanan per unit bahan bakar

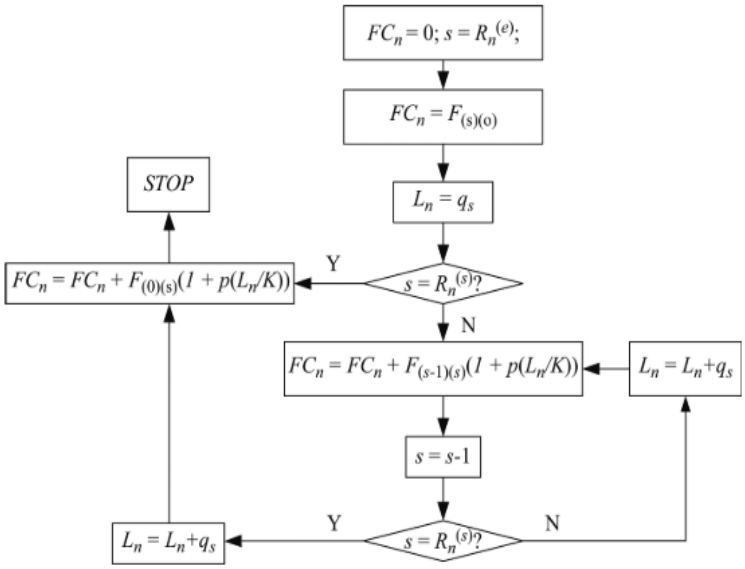
p = kenaikan konsumsi bahan bakar setiap penambahan muatan sebesar 2%

k = kenaikan kapasitas muatan per 100 pound atau 45,35 kilogram.

L_{ij} = beban muatan

Adapun prosedur perhitungan konsumsi bahan bakaer dapat dilihat pada gambar 2.2. Formula perhitungan *fuel consumption* yang digagas oleh Kuo and Wang,

(2011) dilakukan dengan *reversing* urutan dari sub rute yang terbentuk. Gambar 2.2 mendeskripsikan penjumlahan permintaan seluruh pelanggan yang distribusikan dari pelanggan $R_n^{(e)}$ dan diikuti pelangggan selanjutnya secara *reverse* sampai pelanggan $R_n^{(s)}$ dan terakhir yang dikunjungi adalah *distribution center*.



Gambar 2.2 Prosedur perhitungan *fuel consumption*

2.4 Dasar Teori *Particle Swarm Optimization* (PSO)

Particle Swarm Optimization (PSO) adalah Algoritma metaheuristik yang pertama kali digagas oleh Eberhart dan Kennedy pada tahun 1995. PSO merupakan teknik optimasi *stochastic* yang berdasarkan pada tingkah laku sekawanan burung atau sekumpulan ikan dalam bertahan hidup. PSO merupakan salah satu dari teknik komputasi *evolusioner*, dimana populasi pada PSO didasarkan pada pencarian algoritma dan diinisialisasi dengan sebuah populasi acak yang disebut partikel (Trelea, 2003). Menurut Newbould, (2004) PSO memiliki kemiripan dengan *Genetic Algorithm* (GA) yang sistemnya dinisialisasikan dengan populasi solusi acak. Dalam konteks optimasi multivariabel, kawanan disumsikan memiliki ukuran tertentu atau tetap dengan setiap partikel posisi awalnya terletak di suatu lokasi yang acak dalam ruang multidimensi. Setiap partikel diasumsikan memilki dua karakteristik yaitu posisi

dan kecepatan (*velocity*). Setiap partikel bergerak dalam ruang atau space tertentu dan mengingat posisi terbaik yang pernah dilalui atau ditemukan terhadap sumber makanan atau nilai fungsi objektif. Setiap partikel mneyampaikan informasi atau posisi bagusnya kepada partikel yang lain dan menyesuaikan posisi dan kecepatan masing-masing.

Berikut ini merupakan formulasi matematika yang menggambarkan posisi dan kecepatan partikel pada suatu ruang pencarian (Santosa and Willy, 2011) :

$$x_{i(t)} = x_{1(t)}, x_{i(t)}, \dots, x_{iN(t)} \quad (11)$$

$$v_{i(t)} = v_{1(t)}, v_{i(t)}, \dots, v_{iN(t)} \quad (12)$$

Dimana:

x = posisi partikel

v = kecepatan partikel

i = indeks partikel

t = itersi ke-t

N = ukuran dimensi ruang

Berikut ini merupakan model matematika yang menggambarkan pembaruan kecepatan partikel pada setiap iterasi (Santosa and Willy, 2011)

$$v_{i(t+1)} = V_{i(t)} + c_1 r_1 \left(p_{best(i)} - x_{i(t)}\right) + c_2 r_2 \left(g_{best(i)} - x_{1(t)}\right) \quad (13)$$

$$x_{i(t+1)} = v_{i(t+1)} + x_{i(t)} \quad (14)$$

Dimana

$p_{best(i)} = p_{best(i1)}, p_{best(i2)}, p_{best(i3)}, \dots, p_{best(iN)}$ merepresentasikan *local best* dari partikel ke i. Sedangkan $g_{best(i)} = g_{best(i1)}, g_{best(i2)}, g_{best(i3)}, \dots, g_{best(iN)}$ merepresentasikan *global best* dari seluruh partikel. Sedangkan c_1 dan c_2 adalah *coeeficient acceleration* yang bernilai positif. Kemudian r_1 dan r_2 adalah bilangan random yang bernilai antara 0 sampai 1. Persamaan (5) digunakan untuk menghitung kecepatan partikel yang baru berdasarkan kecepatan sebelumnya, jarak antara posisi saat ini dengan posisi terbaik partikel (*local best*), dan jarak antara posisi saat ini dengan posisi terbaik kawanan (*global best*). Kemudian partikel terbang menuju posisi yang baru berdasarkan persamaan (6). Setelah algoritma PSO ini dijalankan dengan sejumlah iterasi tertentu hingga mencapai

kriteria pemberhentian, maka akan diperoleh solusi yang terletak pada *global best*. Jika nilai p_{best} lebih rendah daripada nilai *global best* (g_{best}), maka solusi *local best* menggantikan solusi *global best*.

Kesederhanaan algoritma dan performansinya yang baik, menjadikan PSO telah menarik banyak perhatian di kalangan para peneliti dan telah diaplikasikan dalam berbagai persoalan optimisasi. PSO telah populer menjadi optimisasi global dengan sebagian besar permasalahan dapat diselesaikan dengan baik di mana variabel-variabelnya adalah bilangan riil.

Menurut Bai, (2010) keuntungan dari algoritma PSO adalah:

1. PSO didasarkan pada *intellegence* sehingga dapat diterapkan pada penelitian yang bersifat saintek dan teknik.
2. Pencarian dilakukan oleh kecepatan partikel. Selama proses pencarian, kebanyakan partikel yang optimis yang dapat mengirim informasi ke partikel lain dan kecepatan pencariannya sangat cepat
3. Algortima PSO memiliki perhitungan sederhana, dan mempunyai kemampuan optimasi yang besar sehingga dapat diselesaikan dengan mudah.
4. PSO menggunakan kode/jumlah rill, dan diputuskan secara langsung oleh solusinya. Selain itu jumlah dimensi sama dengan jumlah solusi.

2.5 *Time Varying Accelaration Coefficient* PSO

Persamaan (5) menggambarkan pencarian dalam ruang solusi dipengaruhi oleh dua komponen *stochastic acceleration* (*cognitive component* c_1 dan *social component* c_2). Karena itu, penting untuk mengontrol dua komponen tersebut untuk menentukan solusi optimum secara akurat dan efisien.

Eberhart and Kennedy, (1995) menggambarkan bahwa jika nilai *cognitive component* relative lebih besar dari *social component*, akan menyebabkan partikel melebihi batas ruang pencarian. Sebaliknya, jika nilai *social component lebih besar dari cognitive component* akan membuat pencarian partikel lebih premature sebelum menuju local optimal. Selanjutnya, mereka menetapkan nilai *coefficient acceleration* adalah 2. Nilai *coefficient acceleration* yang seragam akan menyebabkan pencarian partikel dalam ruang hanya setengah waktu pencarian.

Berdasarkan masalah tersebut Ratnaweera et al., (2004) menggagas *time varying acceleration coefficient* sebagai strategi penentuan parameter dalam konsep PSO. Tujuan pengembangan parameter ini adalah memperluas meningkatkan pencarian *global* saat waktu awal optimisasi dan mendorong untuk berkumpul menuju optima global pada akhir pencarian.

Ratnaweera mengurangi nilai *cognitive component* dan meningkatkan, *social component* dengan mengubah *acceleration coefficient* c_1 dan c_2 sepanjang iterasi. Nilai c_1 yang besar dan c_2 lebih kecil di awal, partikel diizinkan untuk bergerak di sekitar ruang pencarian dan bergerak ke arah populasi terbaik. Di sisi lain, nilai *cognitive component* yang kecil dan *social component* yang besar memungkinkan partikel-partikel konvergen menuju global optima di bagian akhir optimasi. Ratnaweera menyarankan metode ini dijalankan dengan faktor bobot inersia yang bervariasi terhadap waktu sesuai pada persamaan (15).

$$\omega_{curr\_iter} = (\omega_{max} - \omega_{min}) * \frac{\max\_iter - curr\_iter}{\max\_iter} + \omega_{min} \quad (15)$$

Nilai $c_1$dan c_2 ditulis sebagi berikut:

$$c_1 = (c_{1max} - c_{1min}) * \frac{curr\_iter}{\max\_iter} + c_{1min} \quad (16)$$

$$c_2 = (c_{2max} - c_{2min}) * \frac{curr\_iter}{\max\_iter} + c_{2min} \quad (17)$$

2.6 Proses Dasar Algoritma PSO

Menurut (Santosa and Willy, 2011) proses implementasi algortima PSO dapat diuraikan sebagai berikut:

1. Asumsikan ukuran kelompok atau kawanan (jumlah partikel) adalah N. Inisialisasi posisi dan kecepatan (*velocity*) pada setiap partikel dalam N-dimensi ditentukan secara acak.
2. Hitung kecepatan dari semua partikel. Semua partikel bergerak menuju titik optimal dengan kecepatan awal diasumsikan sama dengan nol, set iterasi $i = 1$
3. Nilai *fitness* pada setiap partikel diukur menurut fungsi objektif yang ditentukan. Jika nilai *fitness* setiap partikel pada lokasi saat ini lebih baik dari P_{best} , maka P_{best} diatur untuk posisi saat ini.

4. Bandingkan nilai *fitness* partikel dengan g_{best}. Jika g_{best} yang terbaik, maka g_{best} yang di *update*. Sebaliknya jika nilai saat ini lebih baik dari g_{best}, maka tetapkan g_{best} ke nilai posisi partikel saat ini.
5. *Update* kecepatan (*velocity*) dan posisi partikel berdasarkan persamaan (13) dan (12)
6. Cek apakah solusi sekarang sudah konvergen. Jika posisi semua partikel menuju ke satu nilai yang sama, maka ini disebut konvergen. Jika belum, ulangi langkah dengan memperbarui iterasi $i = i + 1$, dengan cara menghitung nilai baru p_{best} dan g_{best}. Proses iterasi ini dilanjutkan sampai semua partikel menuju ke satu titik solusi yang sama. Biasanya akan ditentukan dengan kriteria penghentian (*stopping criteria*), misalnya jumlah selisih solusi sekarang dengan solusi seblumnya sudah sangat kecil.

Beberapa kondisi berhenti yang dapat dipakai dalam *Particle Swarm Optimization* menurut Omran et al., (2005) adalah: - Berhenti ketika jumlah iterasi telah mencapai jumlah iterasi maksimum yang diperbolehkan, berhenti ketika solusi yang diterima ditemukan, Berhenti ketika tidak ada perkembangan setelah beberapa iterasi.